Theoretical Computer Science 410 (2009) 2401–2409

Contents lists available at ScienceDirect

Theoretical Computer Science

journal homepage: www.elsevier.com/locate/tcs

Decimations of languages and state complexity

Dalia Krieger [a], Avery Miller [a,1], Narad Rampersad [a,2], Bala Ravikumar [b], Jeffrey Shallit [a,*]

[a] School of Computer Science, University of Waterloo, Waterloo, Ontario N2L 3G1, Canada

[b] Computer Science Department, 141 Darwin Hall, Sonoma State University, 1801 East Cotati Avenue, Rohnert Park, CA 94928, USA

ARTICLE INFO

In Honor of Sheng Yu's 60th Birthday

Keywords:
Deterministic finite automaton
State complexity
Decimation
Context-free language
Slender language

ABSTRACT

Let the words of a language L be arranged in increasing radix order: $L = \{w_0, w_1, w_2, \ldots\}$. We consider transformations that extract terms from L in an arithmetic progression. For example, two such transformations are even$(L) = \{w_0, w_2, w_4 \ldots\}$ and odd$(L) = \{w_1, w_3, w_5, \ldots\}$. Lecomte and Rigo observed that if L is regular, then so are even(L), odd(L), and analogous transformations of L. We find good upper and lower bounds on the state complexity of this transformation. We also give an example of a context-free language L such that even(L) is not context-free.

1. Introduction

Let $k \geq 1$ and let $\Sigma = \{a_0, a_1, \ldots, a_{k-1}\}$ be a finite alphabet. We put an ordering on the symbols of Σ by defining $a_0 < a_1 < \cdots < a_{k-1}$. This ordering can be extended to the *radix order*[3] on Σ^* by defining $w < x$ if

- $|w| < |x|$, or
- $|w| = |x|$, where $w = a_0 a_1 \cdots a_{n-1}$, $x = b_0 b_1 \cdots b_{n-1}$, and there exists an index r, $0 \leq r < n$ such that $a_i = b_i$ for $0 \leq i < r$ and $a_r < b_r$.

(For words of the same length, the radix order coincides with the lexicographic order.) Thus, given a language $L = \Sigma^*$, we can consider the elements of L in radix order, say

$$L = \{w_0, w_1, w_2, \ldots\},$$

where $w_0 < w_1 < \cdots$.

Let $I \subseteq \mathbb{N}$ be an index set. Given an infinite language L, we let its extraction by I, $L[I]$, denote the elements of L in radix order corresponding to the indices of I, where an index 0 denotes the first element of L. For example, if $L = \{0, 1\}^* = \{\epsilon, 0, 1, 00, 01, 10, 11, \ldots\}$ and $I = \{2, 3, 5, 7, 11, 13, \ldots\}$, the prime numbers, then $L[I] = \{1, 00, 10, 000, 100, 110, \ldots\}$.

In this paper we give a new proof of a result of Lecomte and Rigo [9], which characterizes those index sets that preserve regularity. Next, we determine upper and lower bounds on the state complexity of the transformation that maps a language

* Corresponding author.
E-mail addresses: d2krieger@cs.uwaterloo.ca (D. Krieger), a4miller@cs.toronto.edu (A. Miller), nrampersad@cs.uwaterloo.ca (N. Rampersad), ravi.kumar@sonoma.edu (B. Ravikumar), shallit@graceland.uwaterloo.ca (J. Shallit).

1 Present address: Department of Computer Science, Sandford Fleming Building, University of Toronto, 10 King's College Road, Toronto, Ontario M5S 3G4, Canada.

2 Present address: Department of Mathematics, University of Winnipeg, Winnipeg, Manitoba R3B 2E9, Canada.

3 Sometimes erroneously called the *lexicographic order* in the literature.

0304-3975/$ – see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2009.02.024

to its "decimation" (extraction by an ultimately periodic index set). Finally, answering an open question of Ravikumar, we show that if a language is context-free, its decimation need not be context-free.

We note that our operation is not the same as the related one previously considered by Birget [4], Shallit [15] and Berstel and Boasson [2], which extracts the lexicographically least word of each length from a language. Nor is our operation the same as that introduced in Berstel, Boasson, Carton, Petazzoni, and Pin [3], which filters each word in a language by extracting the letters in the word that occur in positions specified by an index set. (Our operation simply removes words from a language, but does not change the actual words themselves.)

2. Regularity-preserving index sets

Let $I \subseteq \mathbb{N}$ be an index set. We say that I is *ultimately periodic* if there exist integers $r \geq 0$, $m \geq 1$ such that for all $i \in I$ with $i \geq r$ we have $i \in I \Longrightarrow i + m \in I$.

For a language L, we define the (m, r)-*decimation* $\mathrm{dec}_{m,r}(L)$ to be $L[I]$, where $I = \{im + r : i \geq 0\}$. Two particular decimations of interest are $\mathrm{even}(L) = \mathrm{dec}_{2,0}(L)$ and $\mathrm{odd}(L) = \mathrm{dec}_{2,1}(L)$.

We now introduce some notation. Let us assume that our alphabet is $\Sigma = \{a_0, a_1, \ldots, a_{k-1}\}$ with $a_0 < a_1 < \cdots < a_{k-1}$, and for a word $w \in \Sigma^*$, let $F(w)$ be the set of words that are less than w in the radix order, that is, $F(w) = \{x \in \Sigma^* : x < w\}$.

Lemma 1. *We have*

$$F(wa_j) = \{\epsilon\} \cup F(w)\Sigma \cup \{w\}\{a_0, \ldots, a_{j-1}\},$$

and this union is disjoint.

Proof. Suppose $x < wa_j$. Then either $|x| = 0$, which corresponds to the term $\{\epsilon\}$, or $|x| \geq 1$. In this latter case, we can write $x = ya$ for some symbol $a \in \Sigma$. Then either $y < w$, which corresponds to the term $F(w)\Sigma$, or $y = w$, which corresponds to the last term of the union. ■

We now show how to count the number of words accepted by a deterministic finite automaton (DFA) which are, in radix order, less than a given word.

Lemma 2. *Let $A = (Q, \Sigma, \delta, q_0, F)$ be a DFA with n states. For any finite language L, define $M(L)$ to be the matrix such that the entry in row i and column j is the number of words $x \in L$ with $\delta(q_i, x) = q_j$. For $0 \leq l < k$, define $\mathbf{M}_l$ to be the $n \times n$ matrix where the entry in row i and column j is 1 if $\delta(q_i, a_l) = q_j$, and 0 otherwise. Then*

$$M(F(wa_j)) = M(\{\epsilon\}) + M(F(w))(\mathbf{M}_0 + \mathbf{M}_1 + \cdots + \mathbf{M}_{k-1}) + M(\{w\})(\mathbf{M}_0 + \cdots + \mathbf{M}_{j-1}).$$

Proof. By standard results in path algebra and Lemma 1. ■

We now state and prove a theorem that is essentially due to Lecomte and Rigo [9]. (Their proof is somewhat different, and does not explicitly provide the bound on state complexity that is the main focus of this article.)

Theorem 3. *Let $I \subseteq \mathbb{N}$ be an index set. Then $L[I]$ is regular for all regular languages L if and only if I is either finite or ultimately periodic.*

Proof. Suppose $L[I]$ is regular for all regular languages L. Then, in particular, $L[I]$ is regular for $L = a^*$. But $L[I] = \{a^i : i \in I\}$. Then, by a well-known characterization of unary regular languages [11], I is either finite or ultimately periodic.

For the converse, assume that L is regular. If I is finite, then $L[I]$ is trivially regular. Hence assume that I is ultimately periodic. We can then decompose I as the finite union of arithmetic progressions (mod m). Since the class of regular languages is closed under finite union and finite modification, it suffices to show that $L[I]$ is regular for all I of the form $\{jm + r : j \geq 0\}$ where $m \geq 1, 0 \leq r < m$.

Since L is regular, it is accepted by a deterministic finite automaton $A = (Q, \Sigma, \delta, q_0, F)$, where, as usual, Q is a finite nonempty set of states, δ is the transition function, q_0 is the start state, and F is the set of final states. We show how to construct a new DFA A' that accepts $L[I]$ where $I = \{jm + r : j \geq 0\}$.

Let $Q = \{q_0, q_1, \ldots, q_{n-1}\}$. The states of A' are pairs of the form $\langle \mathbf{v}, q \rangle$, where $\mathbf{v}$ is a vector with entries in $\mathbb{Z}/(m)$ and q is a state of Q. The intent is that if we reach the state $\langle \mathbf{v}, q \rangle$ by a path labeled x, then the ith entry of $\mathbf{v}$ counts the number (modulo m) of words $y < x$ that take M from state q_0 to q_i and, further, that $\delta(q_0, x) = q$.

More formally, let $A' = (Q', \Sigma, \delta', q_0', F')$, where the components are defined as follows. For $0 \leq l < k$, define $\mathbf{M}_l$ to be the $n \times n$ matrix where the entry in row i and column j is 1 if $\delta(q_i, a_l) = q_j$, and 0 otherwise. Let $\mathbf{e}_j$ be the vector with a 1 in position j and 0's elsewhere. Let $\mathbf{M} = \sum_{0 \leq l < k} \mathbf{M}_l$. Let

$$\begin{aligned}
Q' &= (\mathbb{Z}/(m))^n \times Q, \\
q_0' &= \langle [0, 0, \ldots, 0], q_0 \rangle, \\
F' &= \{\langle \mathbf{v}, q \rangle : \sum_{\substack{i \\ q_i \in F}} \mathbf{v}[i] \equiv r \pmod{m} \text{ and } q \in F\},
\end{aligned}$$

and

$$\delta'(\langle \mathbf{v}, q_j\rangle, a_i) = \langle \mathbf{v}\mathbf{M} + \mathbf{e}_0 + \mathbf{e}_j(\mathbf{M}_0 + \mathbf{M}_1 + \cdots + \mathbf{M}_{i-1}), \delta(q_j, a_i)\rangle, \tag{1}$$

where the entries in the matrix product are computed over $\mathbb{Z}/(m)$.

It is now clear that $L(A') = \mathrm{dec}_{m,r}(L)$. ■

Corollary 4. *Suppose $m \geq 1$ and $0 \leq r < m$. If L is regular, accepted by an n-state DFA, then the state complexity of $\mathrm{dec}_{m,r}(L)$ is $\leq nm^n$.*

3. The unary case

In the case where $|\Sigma| = 1$, we can improve the upper bound on the state complexity of $\mathrm{dec}_{m,r}(L)$ as follows:

Theorem 5. *If L is defined over a unary alphabet, accepted by an n-state DFA and $m \geq 1$, $0 \leq r < m$, then $\mathrm{dec}_{m,r}(L)$ is accepted by an mn-state DFA.*

Proof. By a well-known result, the DFA for L consists of a "tail" of t states and a "loop" of $n - t$ states. We can then accept $\mathrm{dec}_{m,r}(L)$ using a tail of at most t/m states and a loop of $m(n - t)$ states. ■

There is also a matching lower bound:

Theorem 6. *Let $L = (\mathtt{a}^n)^*$, accepted by an n-state DFA. Then $\mathrm{dec}_{m,0}(L) = (\mathtt{a}^{mn})^*$, which is accepted by no DFA with less than mn states.*

Proof. Clear. ■

4. Lower bound

We now turn to the question of a lower bound on the state complexity of decimation in the case of larger alphabets.

We introduce some notation. Let $|x|_a$ be the number of occurrences of the symbol a in the word x. For integer $n \geq 1$, define

$$L_n := \{x \in \{0, 1\}^* : |x|_1 \equiv 0 \pmod n\}.$$

Let $\Sigma = \{0, 1\}$. Then L_n can be accepted in the obvious way by a DFA $A_n = (Q, \Sigma, \delta, q_0, F)$ with n states. Here $Q = \{q_0, q_1, \ldots, q_{n-1}\}$, $F = \{q_0\}$, and δ is defined by $\delta(q_i, 0) := q_i$ and $\delta(q_i, 1) := q_{(i+1) \bmod n}$ for $0 \leq i < n$. Note that $\delta(q_0, w) = q_i$ if and only if $|w|_1 \equiv i \pmod n$.

We will prove

Theorem 7. *For odd integers $n \geq 3$, any DFA accepting $\mathrm{odd}(L_n)$ has at least $(n + 1)2^{n-1}$ states.*

The outline of the proof is as follows. First, we use the construction of Theorem 3 to create a DFA A'_n with $n \cdot 2^n$ states accepting $\mathrm{odd}(L_n)$. We then re-interpret the transition function in the case of L_n using Eq. (1). Next, we show that each state of A'_n is reachable from q'_0. Finally, we determine all pairs of equivalent states in A'_n and show that A'_n has $(n+1)2^{n-1}$ pairwise inequivalent states. The result follows by the Myhill–Nerode theorem.

For the rest of this section, we adopt the following conventions. Vectors are denoted in boldface, such as $\mathbf{v}$. Since the vectors we will deal with are in $(\mathbb{Z}/(2))^n$, we write $\mathbf{v} = [v_0, v_1, \ldots, v_{n-1}]$, and arithmetic with vectors and terms of vectors is always done implicitly mod 2. Similarly, any state q_i represents $q_{i \bmod n}$, and we do not explicitly write the (mod n) part. For the basis vector $\mathbf{e}_j$ we write $\mathbf{e}_j = (e_{j,0}, e_{j,1}, \ldots, e_{j,n-1})$.

Lemma 8. *We have $A'_n = (Q', \Sigma, \delta', q'_0, F')$ where*

- $Q' = \{\langle \mathbf{v}, q\rangle : \mathbf{v} \in (\mathbb{Z}/(2))^n, q \in Q\}$;
- $\Sigma = \{0, 1\}$;
- $q'_0 = \langle [0, 0, \ldots, 0], q_0\rangle$;
- $F' = \{\langle \mathbf{v}, q_0\rangle : v_0 = 1\}$;
- $\delta'(\langle [v_0, v_1, \ldots, v_{n-1}], q_i\rangle, 0) = \langle [v_0 + v_{n-1} + 1, v_0 + v_1, v_1 + v_2, \ldots, v_{n-2} + v_{n-1}], q_i\rangle$;
- $\delta'(\langle [v_0, v_1, \ldots, v_{n-1}], q_i\rangle, 1) = \langle \mathbf{e}_i + [v_0 + v_{n-1} + 1, v_0 + v_1, v_1 + v_2, \ldots, v_{n-2} + v_{n-1}], q_{i+1}\rangle$.

Proof. Follows directly from the characterization in Theorem 3. ■

For $a \in \{0, 1\}$, we define $\overline{a} := 1 - a$. If $\mathbf{v} = [v_0, v_1, \ldots, v_{n-1}]$, then

$$\overline{\mathbf{v}} := [\overline{v_0}, \overline{v_1}, \ldots, \overline{v_{n-1}}] = \mathbf{v} + [1, 1, \ldots, 1].$$

If $q = \langle \mathbf{v}, q_i\rangle$, define $\overline{q} := \langle \overline{\mathbf{v}}, q_i\rangle$.

We say that the *parity* of $\langle \mathbf{v}, q_i\rangle$ is odd if $\mathbf{v}$ contains an odd number of entries equal to 1. Otherwise the parity of $\langle \mathbf{v}, q_i\rangle$ is even.

Lemma 9. *For all $q \in Q'$, the parity of $\delta'(q, 0)$ is odd and the parity of $\delta'(q, 1)$ is even.*

Proof. From Lemma 8 we have that the sum of the entries of $\delta'(q, 0)$ is $2v_0 + 2v_1 + \cdots + 2v_{n-1} + 1 \equiv 1 \pmod 2$, and the sum of the entries of $\delta'(q, 1)$ is $2v_0 + 2v_1 + \cdots + 2v_{n-1} + 1 + e_{i,i} \equiv 0 \pmod 2$. ■

Lemma 10. *Let $p, q \in Q'$. Then $\delta'(p, s) = \delta'(q, s)$ for some $s \in \Sigma^*$ iff $p = q$ or $p = \overline{q}$.*

Proof. If $p = q$, then $\delta'(p, \epsilon) = \delta'(q, \epsilon)$. Hence assume that $p = \overline{q}$. It now immediately follows from Lemma 8 that $\delta'(p, a) = \delta'(q, a)$ for $a \in \{0, 1\}$.

Now we prove the converse. Suppose $\delta'(p, s) = \delta'(q, s)$ for some $s \in \Sigma^*$. If $p = q$ we are done, so we may assume that $p \neq q$. Let $p = \langle \mathbf{v}, q_i \rangle$ and $q = \langle \mathbf{w}, q_j \rangle$. Then from $\delta'(p, s) = \delta'(q, s)$ we get $q_i = q_j$.

Let t be the shortest prefix of s such that $\delta'(p, t) = \delta'(q, t)$. If $t = \epsilon$ then $p = q$, a contradiction. Hence $|t| \geq 1$.

Case 1: $|t| = 1$. Suppose $t = 0$. From Lemma 8, we deduce that if $\langle \mathbf{u}, r \rangle = \delta'(\langle \mathbf{v}, q_i \rangle, 0)$ then $r = q_i$ and

$$\begin{aligned} u_0 &= v_0 + v_{n-1} + 1 \\ u_1 &= v_0 + v_1 \\ u_2 &= v_1 + v_2 \\ &\vdots \\ u_{n-1} &= v_{n-2} + v_{n-1}. \end{aligned}$$

Hence

$$\begin{aligned} v_{n-1} &= u_0 + v_0 + 1 \\ v_{n-2} &= v_{n-1} + u_{n-1} = u_{n-1} + u_0 + v_0 + 1 \\ v_{n-3} &= u_{n-2} + u_{n-1} + u_0 + v_0 + 1 \\ &\vdots \\ v_1 &= u_2 + v_2 = u_2 + u_3 + \cdots + u_{n-1} + u_0 + v_0 + 1. \end{aligned}$$

Thus

$$\mathbf{v} = [v_0, v_0, v_0, \ldots, v_0] + [0, u_2 + \cdots + u_{n-1} + u_0 + 1, u_3 + \cdots + u_{n-1} + u_0 + 1, \ldots, u_{n-1} + u_0 + 1, u_0 + 1].$$

Similarly,

$$\mathbf{w} = [w_0, w_0, w_0, \ldots, w_0] + [0, u_2 + \cdots + u_{n-1} + u_0 + 1, u_3 + \cdots + u_{n-1} + u_0 + 1, \ldots, u_{n-1} + u_0 + 1, u_0 + 1].$$

Since $\mathbf{v} \neq \mathbf{w}$, it follows that $v_0 \neq w_0$. Thus $v_0 = \overline{w_0}$ and $\mathbf{v} = \overline{\mathbf{w}}$. Hence $p = \overline{q}$.

On the other hand, if $t = 1$, then similar reasoning gives $q = q_{i+1}$ and

$$\begin{aligned} \mathbf{v} &= v_0 + [0, u_2 + \cdots + u_{n-1} + u_0 + 1, u_3 + \cdots + u_{n-1} + u_0 + 1, \ldots, u_{n-1} + u_0 + 1, u_0 + 1] \\ &\quad + [0, e_{i,2} + \cdots + e_{i,n-1} + e_{i,0}, e_{i,3} + \cdots + e_{i,n-1} + e_{i,0}, \ldots, e_{i,n-1} + e_{i,0}, e_{i,0}] \end{aligned}$$

and

$$\begin{aligned} \mathbf{w} &= w_0 + [0, u_2 + \cdots + u_{n-1} + u_0 + 1, u_3 + \cdots + u_{n-1} + u_0 + 1, \ldots, u_{n-1} + u_0 + 1, u_0 + 1] \\ &\quad + [0, e_{i,2} + \cdots + e_{i,n-1} + e_{i,0}, e_{i,3} + \cdots + e_{i,n-1} + e_{i,0}, \ldots, e_{i,n-1} + e_{i,0}, e_{i,0}]. \end{aligned}$$

Again, since $\mathbf{v} \neq \mathbf{w}$, it follows that $v_0 \neq w_0$. Thus $v_0 = \overline{w_0}$ and $\mathbf{v} = \overline{\mathbf{w}}$. Thus $p = \overline{q}$.

Case 2: $|t| > 1$. Write $t = rab$ for $a, b \in \Sigma$, $r \in \Sigma^*$. Let $p' = \delta'(p, ra)$ and $q' = \delta'(q, ra)$. Then $p' \neq q'$ by definition of t and r. However, $\delta'(p', b) = \delta'(q', b)$, so from Case 1 we have $p' = \overline{q'}$. But then, since n is odd, the parities of p' and q' differ.

On the other hand, $p' = \delta'(\delta'(p, r), a)$ and $q' = \delta'(\delta'(q, r), a)$. From Lemma 9, we conclude that p' and q' are of the same parity. This is a contradiction, and so this case cannot occur. ■

Corollary 11. *In the transition diagram of A', every state p has exactly two incoming arrows, both labeled with the same letter a, arising from states of different parity, q and $\overline{q}$. If p is of odd parity, then $a = 0$, and if p is of even parity, then $a = 1$.*

Proof. This follows from the proof of Lemma 10, where $|s| = 1$. ■

We say that a state $q \in Q'$ is reachable if there exists a string $x \in \{0, 1\}^*$ such that $\delta'(q'_0, x) = q$.

Lemma 12. *Every state of A' is reachable.*

Proof. Here is the outline of the proof. We define two partial functions:

$$\text{INCR} : \{0,1\}^* \times \{0,1,\ldots,n-1\} \times \{0,1,\ldots,n-1\} \to \{0,1\}^*$$
$$\text{SHIFT} : \{0,1\}^* \times \{0,1,\ldots,n-1\} \to \{0,1\}^*.$$

INCR(t, k, l) produces a string t' such that if $\delta'(q'_0, t) = \langle \mathbf{w}, q_j\rangle$ and $\mathbf{w}$ has odd parity, then $\delta'(q'_0, tt') = \langle \mathbf{w} + \mathbf{e}_k + \mathbf{e}_l, q_l\rangle$. In other words, the effect of reading t' after t has been read is to increment the kth and lth bits in the first component of the state, and change the second component to q_l.

SHIFT(t, l) produces a string t' such that if $\delta'(q'_0, t) = \langle \mathbf{w}, q_j\rangle$ and $\mathbf{w}$ has odd parity, then $\delta'(q'_0, tt') = \langle \mathbf{w}, q_l\rangle$. In other words, the effect of reading t' after t has been read is to change the second component of the state to q_l.

We will show below how to define these two functions. For the moment, however, assume that these functions exist; we show how to apply them successively to form a path to any state $\langle \mathbf{v}, q_i\rangle$. The general idea is to apply INCR to add 1-bits to the first component of the state, and then fix up the second component by applying SHIFT.

We start with $t = 0$; this takes us from q'_0 to the state $\langle [1, 0, 0, \ldots, 0], q_0\rangle$.

Case 1: $\langle \mathbf{v}, q_i\rangle$ has odd parity. Find the minimum index l such that $v_l = 1$. If $l = 0$, then no action is necessary. If $l \neq 0$, use INCR$(t, 0, l)$ to get to the state $\langle \mathbf{e}_l, q_l\rangle$. At this point the first 1-bit is set correctly. Since $\mathbf{v}$ has odd parity, there is an even number, say $2j$, of remaining 1-bits. We now apply INCR j times to increment the remaining 1-bits in pairs. Because we change an even number of bits each time, each new state reached after an application of INCR will be of odd parity. Finally, fix up the second component by applying SHIFT.

Case 2: $p = \langle \mathbf{v}, q_i\rangle$ has even parity. By Corollary 11 there is a unique state $q = \langle \mathbf{u}, q_{i-1}\rangle$ of odd parity such that $\delta'(q, 1) = p$. Use Case 1 to get to q, and then append 1 to get to p.

It now remains to see how to construct the functions INCR and SHIFT.

First, we show that from any reachable state with odd parity, we eventually return to that state after reading some number of 0's.

Lemma 13. *Given a state of odd parity p, and any word $s \in \{0,1\}^*$ such that $\delta'(q'_0, s) = p$, there exists $t = 0^l$, $l \geq 1$, such that $\delta'(q'_0, st) = p$.*

Proof. Using Lemma 9, we know that the parity of each of the states $\delta'(q'_0, s0^i)$, $i \geq 0$, is odd. Since there are only a finite number of states, we must have $r := \delta'(q'_0, s0^i) = \delta'(q'_0, s0^j)$ for some $0 \leq i < j$. Further, choose i to be minimal and j to be minimal for this i. Suppose, to get a contradiction, that $i \geq 1$. Define $r' := \delta'(q'_0, s0^{i-1})$ and $r'' := \delta'(q'_0, s0^{j-1})$. Then $r' \neq r''$, for otherwise i, j would not be minimal. Then r' and r'' are distinct states of odd parity from which we reach r on input 0, contradicting Corollary 11. Hence $i = 0$, and we can take $l = j$. ■

Now let p be a reachable state of odd parity. Let $l(p)$ be the least positive integer l such that $\delta'(p, 0^l) = p$.

Lemma 14. *If $p = \langle \mathbf{v}, q_i\rangle$ is a reachable state of odd parity, then $l(p) \geq 3$ unless $\mathbf{v} = [0, 0, 0, \ldots, 1]$, in which case $l(p) = 1$.*

Proof. If $\mathbf{v} = [0, 0, 0, \ldots, 1]$, then from Lemma 8 we get $\delta'(\langle \mathbf{v}, q_i\rangle, 0) = \langle \mathbf{v}, q_i\rangle$, so $l(p) = 1$.

For the converse, suppose $l(p) = 1$. Then if $\mathbf{v} = [v_0, v_1, \ldots, v_{n-1}]$, we get by Lemma 8 that

$$[v_0, v_1, \ldots, v_{n-1}] = [v_0 + v_{n-1} + 1, v_0 + v_1, v_1 + v_2, \ldots, v_{n-2} + v_{n-1}].$$

Solving this system gives $\mathbf{v} = [0, 0, \ldots, 1]$.

If $l(p) = 2$, then by Lemma 8 we get

$$[v_0, v_1, \ldots, v_{n-1}] = [v_0 + v_{n-2}, v_1 + v_{n-1} + 1, v_0 + v_2, v_1 + v_3, \ldots, v_{n-3} + v_{n-1}].$$

Solving this system gives $\mathbf{v} = [0, 0, \ldots, 1]$; but then $l(p) = 1$, a contradiction. ■

We now define $\tau(p) := \max(3, l(p))$; hence if p is a reachable state of odd parity, then $\tau(p) \geq 3$ and $\delta'(p, 0^{\tau(p)}) = p$.

Lemma 15. *Let $p = \langle \mathbf{v}, q_i\rangle$ be a reachable state of odd parity. Then*

(a) $\delta'(p, 0^{\tau(p)-3}010) = \langle \mathbf{v} + \mathbf{e}_i + \mathbf{e}_{i+1}, q_{i+1}\rangle$;
(b) $\delta'(p, 0^{\tau(p)-3}110) = \langle \mathbf{v} + \mathbf{e}_i + \mathbf{e}_{i+1}, q_{i+2}\rangle$.

Proof. Since p is reachable, there exists a string s such that $\delta'(q'_0, s) = p = \langle \mathbf{v}, q_i\rangle$. Now $\delta'(q'_0, s) = \delta'(q'_0, s0^{\tau(p)})$. From the construction of A' we know that if $\mathbf{v} = [v_0, v_1, \ldots, v_{n-1}]$ then v_i counts, modulo 2, the number n of words w such that w is lexicographically less than $s0^{\tau(p)}$ and $|w|_1 \equiv i \pmod n$. Now consider the words from $s0^{\tau(p)}$ to $s0^{\tau(p)-3}110$. In increasing lexicographic order, they are

$s0^{\tau(p)-3}000$
$s0^{\tau(p)-3}001$
$s0^{\tau(p)-3}010$
$s0^{\tau(p)-3}011$
$s0^{\tau(p)-3}100$
$s0^{\tau(p)-3}101$
$s0^{\tau(p)-3}110$.

Now $|s|_1 = |s0^{\tau(p)-3}000|_1 \equiv i \pmod n$. Thus

$$\delta'(q'_0, s0^{\tau(p)-3}001) = \langle \mathbf{v} + \mathbf{e}_i, q_{i+1}\rangle.$$

Similarly, $|s0^{\tau(p)-3}001| \equiv i+1 \pmod n$. Thus

$$\delta'(q'_0, s0^{\tau(p)-3}010) = \langle \mathbf{v} + \mathbf{e}_i + \mathbf{e}_{i+1}, q_{i+1}\rangle.$$

Thus (a) is proved.

With a similar computation, we find

$$\delta'(q'_0, s0^{\tau(p)-3}110) = \langle \mathbf{v} + \mathbf{e}_i + \mathbf{e}_{i+1}, q_{i+2}\rangle.$$

This proves (b). ■

Corollary 16. *Let $p = \langle \mathbf{v}, q_i\rangle$ be any reachable state of odd parity in A'. For all $k \geq 1$, there exists a word $y_k \in \{0,1\}^*$ such that $\delta'(p, y_k) = \langle \mathbf{v} + \mathbf{e}_i + \mathbf{e}_{i+k}, q_{i+k}\rangle$.*

Proof. Using Lemma 15(a), we have $\delta'(p, 0^{\tau(p)-3}010) = \langle \mathbf{v}+\mathbf{e}_i+\mathbf{e}_{i+1}, q_{i+1}\rangle$. If $k = 1$, we are done. Otherwise, use induction. Suppose we have found a string x_k such that $\delta'(p, x_k) = p' := \langle \mathbf{v} + \mathbf{e}_i + \mathbf{e}_{i+k-1}, q_{i+k-1}\rangle$. Then by Lemma 15(a) we have $\delta'(p', 0^{\tau(p')-3}010) = \langle \mathbf{v} + \mathbf{e}_i + \mathbf{e}_{i+k}, q_{i+k}\rangle$. Thus we can take $y_k = x_k 0^{\tau(p')-3}010$. ■

Now let us show that the function SHIFT exists.

Lemma 17. *Let $p = \langle \mathbf{v}, q_i\rangle$ be any reachable state of odd parity in A'. Then for all $j \geq 0$ there exists a word $w_j \in \{0,1\}^*$ such that $\delta'(p, w_j) = \langle \mathbf{v}, q_j\rangle$.*

Proof. If $i = j$ we can take $w_j = \epsilon$.

Otherwise, use Corollary 16 with $k = n-2$ to get to the state $\langle \mathbf{v} + \mathbf{e}_i + \mathbf{e}_{i+n-2}, q_{i+n-2}\rangle$. Now use Lemma 15 (b) to get to state $\langle \mathbf{v} + \mathbf{e}_i + \mathbf{e}_{i+n-1}, q_i\rangle$. Now use Corollary 16 with $k = n-1$ to get to the state $\langle \mathbf{v}, q_{i-1}\rangle$. If $j \equiv i-1 \pmod n$, we are done. Otherwise, repeat the sequence of steps above until j is reached. ■

Thus the SHIFT function exists. We now turn to INCR.

Lemma 18. *Let $p = \langle \mathbf{v}, q_i\rangle$ be any reachable state of odd parity in A'. Then there exists a word $x_{j,l} \in \{0,1\}^*$ such that $\delta'(p, x_{j,l}) = \langle \mathbf{v} + \mathbf{e}_j + \mathbf{e}_l, q_l\rangle$.*

Proof. First, use SHIFT to get to the state $\langle \mathbf{v}, q_j\rangle$. From there, use Corollary 16 with $k = l - j$ to get to state $\langle \mathbf{v} + \mathbf{e}_j + \mathbf{e}_l, q_l\rangle$. This shows that INCR exists. ■

We have now completed the proof of Lemma 12. ■

Now that we know that every state of A' is reachable, it remains to show that the number of pairwise distinguishable states is $(n+1)2^{n-1}$.

To do so, we determine when two states are equivalent. We say that a state p is equivalent to q if, for all $x \in \Sigma^*$, we have $\delta'(p, x) \in F'$ iff $\delta'(q, x) \in F'$.

The first step is the following lemma.

Lemma 19. *Let $p_0, r_0 \in Q'$. Suppose there exists a word $s \in \Sigma^*$ such that $\delta'(p_0, s) = p_1$ and $\delta'(r_0, s) = r_1$ where $p_1 \neq r_1$ and $p_1, r_1 \in F'$. Then there exists a word $t = 0^k$, $k \geq 1$, such that exactly one of $\{\delta'(p_1, t), \delta'(r_1, t)\}$ is in F'.*

Proof. Since $p_1, r_1 \in F'$, we can write

$$\begin{aligned} p_1 &= \langle [u_0, u_1, \ldots, u_{n-1}], q_0\rangle \\ r_1 &= \langle [v_0, v_1, \ldots, v_{n-1}], q_0\rangle, \end{aligned}$$

where $u_0 = v_0 = 1$. Let i be the greatest index such that $u_i \neq v_i$; since by hypothesis $p_1 \neq r_1$, such an index must exist, and since $u_0 = v_0 = 1$, we have $1 \leq i \leq n-1$. By the definition of i we have $\overline{u_i} = v_i$ and $u_j = v_j$ for $j > i$. Define $p_2 := \delta'(p_1, 0)$ and $r_2 := \delta'(r_1, 0)$.

Suppose $i = n-1$. Then from Lemma 8 we have

$$\begin{aligned} p_2 &= \langle [u_0 + u_{n-1} + 1, u_0 + u_1, u_1 + u_2, \ldots, u_{n-2} + u_{n-1}], q_0\rangle \\ r_2 &= \langle [v_0 + v_{n-1} + 1, v_0 + v_1, v_1 + v_2, \ldots, v_{n-2} + v_{n-1}], q_0\rangle. \end{aligned}$$

Consider the first entries of the vectors in p_2 and r_2. Since $\overline{u_{n-1}} = v_{n-1}$, we get that $v_0 + v_{n-1} + 1 = v_0 + \overline{u_{n-1}} + 1$. Since $u_0 = v_0 = 1$, this differs from $u_0 + u_{n-1} + 1$. Thus at most one of p_2, r_2 is in F', and the conclusion follows with $t = 0$, $k = 1$.

Otherwise $i < n-1$. Write

$$\begin{aligned} p_2 &= \langle [x_0, \ldots, x_{n-1}], q_0\rangle \\ r_2 &= \langle [y_0, \ldots, y_{n-1}], q_0\rangle. \end{aligned}$$

We have $\overline{u_i} = v_i$ and $u_{i+1} = v_{i+1}$. Also, $1 \le i \le n-2$, so $2 \le i+1 \le n-1$. Now by Lemma 8 we get $x_{i+1} = u_i + u_{i+1}$ and

$$\begin{aligned} y_{i+1} &= v_i + v_{i+1} \\ &= \overline{u_i} + u_{i+1}, \end{aligned}$$

so it follows that $x_{i+1} = \overline{y_{i+1}}$. Thus the largest index j where $x_j \neq y_j$ is $\ge i+1$. We now repeat this process until $j = n-1$, at which point we can finish with the argument above. ■

Next we show that we can always get to at least one final state from any state.

Lemma 20. *At least one final state of A' is reachable from any state of A'.*

Proof. Let $p = \langle \mathbf{v}, q_i \rangle$ be a state of A'. From Lemma 12 we know that there is a string y such that $\delta'(q'_0, y) = p$. Now let $s_1 = 1^{n-1-i}01$ and $s_2 = 1^{n-1-i}10$. Clearly ys_2 directly follows ys_1 in lexicographic order, and both $ys_1, ys_2 \in L$. So at least one of these two strings must be in $\mathrm{odd}(L)$. ■

We now consider when two distinct states $p = \langle \mathbf{v}, q_i \rangle$ and $q = \langle \mathbf{w}, q_j \rangle$ are equivalent.

Lemma 21. *A state $p = \langle \mathbf{v}, q_i \rangle$ is equivalent to $q = \langle \mathbf{w}, q_j \rangle$ iff $p = \overline{q}$ and $i = j \neq 0$.*

Proof. By Lemma 20 we know that there is a word s such that $\delta'(p, s) = f_1 \in F'$. If $\delta'(q, s) \notin F'$, then p and q are inequivalent. Thus assume that $\delta'(q, s) = f_2 \in F'$. If $f_2 \neq f_1$, then we use Lemma 19 to see that f_1 and f_2 are not equivalent. Thus p and q are not equivalent.

It follows that $f_1 = f_2$. Hence $i = j$. Thus $\delta'(p, s) = \delta'(q, s)$. By Lemma 10, we know that $p = \overline{q}$. If $i = 0$, then p and q are inequivalent, since the string ϵ distinguishes them ($v_0 = \overline{w_0}$, so exactly one of these is 1). If $i \neq 0$, then we claim p and q are equivalent. To do so, we consider $\delta'(p, t)$ and $\delta'(q, t)$ for all strings t.

If $|t| = 0$, then neither $\delta'(p, t) = p$ nor $\delta'(q, t) = q$ is in F', since in order to be in F' a state's second component must be q_0.

If $|t| = 1$, then from Lemma 8 and the fact that $p = \overline{q}$, we see that $\delta'(q, t) = \delta'(p, t)$. From this we see immediately that $\delta'(q, u) = \delta'(p, u)$ for all $|u| \ge 2$.

Thus the result follows. ■

Lemma 22. *The number of pairwise distinguishable states is $n \cdot 2^n - (n-1)2^{n-1} = (n+1)2^{n-1}$.*

Proof. There are $n \cdot 2^n$ states in A'_n. These are all reachable by Lemma 12. Of this number, a state is equivalent to at most one other state, and this occurs iff the state is of the form $\langle \mathbf{v}, q_i \rangle$ with $i \neq 0$. Thus we need to subtract $(n-1)2^{n-1}$ to account for the equivalent states, leaving $(n+1)2^{n-1}$ pairwise inequivalent states. ■

We have now completed the proof of Theorem 7.

5. Decimations of context-free languages

Suppose L is a context-free language. In some cases, decimations of L are still context-free. For example, if $PAL = \{x \in \{\mathtt{a}, \mathtt{b}\}^* : x = x^R\}$, the palindrome language, then $\mathrm{even}(L) = \{\epsilon\} \cup \{x\mathtt{b}x^R : x \in \{\mathtt{a}, \mathtt{b}\}^*\} \cup \{x\mathtt{bb}x^R : x \in \{\mathtt{a}, \mathtt{b}\}^*\}$, which is clearly context-free. If $L = \{\mathtt{a}^n\mathtt{b}^n : n \ge 0\}$, then it is easy to see that *any* decimation of L is context-free.

This raises the following natural question: if L is a context-free language (CFL), need its decimation be context-free? In this section we give two examples where this is *not* the case.

For the first example, let B be the balanced parentheses language on the symbols $\{\mathtt{a}, \mathtt{b}\}$, i.e.,

$$B = \{\epsilon, \mathtt{ab}, \mathtt{aabb}, \mathtt{abab}, \mathtt{aaabbb}, \mathtt{aababb}, \mathtt{aabbab}, \mathtt{abaabb}, \mathtt{ababab}, \mathtt{aaaabbbb}, \ldots\}.$$

This is a well-known CFL, generated by the context-free grammar

$$S \to \mathtt{a}S\mathtt{b}S \mid \epsilon.$$

We will show that $\mathrm{even}(B) = \{\epsilon, \mathtt{aabb}, \mathtt{aaabbb}, \mathtt{aabbab}, \mathtt{ababab}, \ldots\}$ is not a CFL.

First, we state some useful lemmas.

Lemma 23. *The number of words of length $2n$ in B is the Catalan number $C_n = \binom{2n}{n}/(n+1)$.*

Proof. Very well known; for example, see [10, pp. 116–117]. ■

Now let $\nu_2(n)$ denote the exponent of the highest power of 2 dividing n, and let $s_2(n)$ denote the number of 1's in the binary expansion of n.

Lemma 24. *For $n \ge 0$ we have $\nu_2(n!) = n - s_2(n)$.*

Proof. A well-known result due to Legendre; for example, see [1, Corollary 3.2.2]. ■

Lemma 25. *For $n \geq 0$, C_n is odd if and only if $n = 2^i - 1$ for some integer $i \geq 0$.*

Proof. We have

$$\begin{aligned}\nu_2(C_n) &= \nu_2\left(\frac{\binom{2n}{n}}{n+1}\right)\\ &= \nu_2((2n)!) - 2\nu_2(n!) - \nu_2(n+1)\\ &= (2n - s_2(2n)) - 2(n - s_2(n)) - \nu_2(n+1)\\ &= s_2(n) - \nu_2(n+1).\end{aligned}$$

Thus C_n is odd if and only if $s_2(n) = \nu_2(n+1)$, if and only if $n = 2^i - 1$ for some $i \geq 0$. ■

Lemma 26. *For $n \geq 0$ define $D_n := \sum_{1 \leq i \leq n} C_n$. (Thus $D_0 = 0$.) Then D_n is even if and only if there exists $i \geq 0$ such that $2^{2i} - 1 \leq n < 2^{2i+1} - 1$.*

Proof. Follows immediately from Lemma 25. ■

We are now ready to prove

Theorem 27. *The language* even(B) *is not a context-free language.*

Proof. First, we observe that $(\mathtt{ab})^n$ is the lexicographically greatest word of length $2n$ in B. It follows that $(\mathtt{ab})^n$ is the $D_n = (\sum_{1 \leq i \leq n} C_i)$th word in B in the radix order. (Recall that we start indexing at 0.)

Suppose even(B) is context-free, and define the morphism $h : \{\mathtt{c}\}^* \to \{\mathtt{a}, \mathtt{b}\}^*$ by $h(\mathtt{c}) = \mathtt{ab}$. By a well-known theorem [6, Theorem 6.3], $h^{-1}(\text{even}(B))$ is a context-free language. But $h^{-1}(\text{even}(B)) = \{\mathtt{c}^n : D_n \text{ is even}\}$. From Lemma 26, we have

$$h^{-1}(\text{even}(B)) = \{\mathtt{c}^n : \exists i \geq 0 \text{ such that } 2^{2i} - 1 \leq n < 2^{2i+1} - 1\}.$$

Since $h^{-1}(\text{even}(B))$ is a unary CFL, by a well-known theorem it is actually regular. But the lengths of strings in a unary regular language form an ultimately periodic set, a contradiction. Hence even(B) is not context-free. ■

Corollary 28. odd(B) *is not context-free.*

Proof. This follows from the fact that $h^{-1}(\text{odd}(B)) = \mathtt{c}^* - h^{-1}(\text{even}(B))$. ■

Recall that a language is a deterministic context-free language (DCFL) if it is accepted by a pushdown automaton that has at most one choice for a move from every configuration.

Corollary 29. *The class of DCFL's is not closed under decimation.*

Proof. B is a DCFL, and even(B) is not a CFL. ■

For our second example, consider the language

$$\begin{aligned}D &= \{x \in \{\mathtt{a}, \mathtt{b}\}^* : |x|_{\mathtt{a}} = |x|_{\mathtt{b}}\}\\ &= \{\epsilon, \mathtt{ab}, \mathtt{ba}, \mathtt{aabb}, \mathtt{abab}, \mathtt{abba}, \mathtt{baab}, \mathtt{baba}, \mathtt{bbaa}, \mathtt{aaabbb}, \ldots\}.\end{aligned}$$

We will show

Theorem 30. even(D) *is not context-free.*

Proof. The proof is similar to that for the language B. We assume that even(D) is context-free and get a contradiction.

First, note that there are $\binom{n}{n/2}$ strings of length n in D if n is even, and 0 if n is odd. In particular, the number of strings of length n in D is even for $n > 0$. Since D contains the empty string, a nonempty string w is in even(D) if and only if it is of odd index, lexicographically speaking, among the strings of length n in D.

Since, by assumption, even(D) is context-free, so is

$$D' = \text{even}(D) \cap \mathtt{aba}^*\mathtt{b}^* = \{\mathtt{abab}, \mathtt{abaaabbb} \ldots\}.$$

We claim that $\mathtt{aba}^n\mathtt{b}^n$ is, lexicographically speaking, of index $\binom{2n}{n-1}$ among all strings in D of length $2n + 2$. To see this, observe that a string of length $2n + 2$ is lexicographically less than $\mathtt{aba}^n\mathtt{b}^n$ if and only if it begins with $\mathtt{aa}$.

Thus $\mathtt{aba}^n\mathtt{b}^n \in$ even(D) if and only if $\binom{2n}{n-1}$ is odd. Now $\binom{2n}{n-1}$ is odd if and only if $n = 2^k - 1$ for some $k \geq 1$. Thus $D' = \{\mathtt{aba}^{2^k-1}\mathtt{b}^{2^k-1} : k \geq 1\}$, which is clearly not context-free. ■

6. Decimation and slender languages

Next we consider extractions and decimations of slender context-free languages. A language L is *slender* if there exists a constant c such that for every $n \geq 0$, the number of words of length n in L is $\leq c$. Charlier, Rigo, and Steiner [5] showed that if L is regular and slender, then extraction by an index set I gives a regular language if and only if I is the finite union of arithmetic progressions. We will show that the class of slender context-free languages is closed under the operation $\mathrm{dec}_{m,r}$.

We first review some properties of slender context-free languages. Ilie [7,8], confirming a conjecture of Păun and Salomaa [12], proved that a context-free language is slender if and only if it is a finite disjoint union of languages of the form $\{uv^n wx^n y : n \geq 0\}$, and further, such a decomposition is effectively computable. Ilie [8, Corollary 13] also proved that the class of slender context-free languages is effectively closed under intersection and set difference.

Theorem 31. *The class of slender context-free languages is effectively closed under the operation $\mathrm{dec}_{m,r}$.*

Proof. Let L be a slender context-free language and let c be an upper bound on the number of words of any given length in L. We write L as a finite union $L = \bigcup_{i=1}^{c} L_i$, where, for $i = 1, \ldots, c$, L_i is the set consisting of the lexicographically ith words of each length in L. We first show that each L_i is context-free.

Let $\min(L)$ denote the set of the lexicographically least words of each length in L. Berstel and Boasson [2] showed that for any context-free language L, $\min(L)$ is context-free, and further, this closure is effective. In our case, the language L is slender by assumption, and the language $L_1 = \min(L)$ is slender by definition. Since the class of slender context-free languages is closed under set difference, we see that the language $L' = L \setminus L_1$ is also a slender context-free language. We next define $L_2 := \min(L')$. Continuing this process, we see that each L_i is a slender context-free language, as required, and further, this decomposition is effectively computable.

For $i = 1, \ldots, c$, let A_i be a PDA accepting L_i. We show how to accept $\mathrm{dec}_{m,r}(L)$ by modifying each A_i appropriately. Recall that we may write L as a finite disjoint union $L = \bigcup_{j=1}^{k} P_j$, where each P_j is a language of the form $\{uv^n wx^n y : n \geq 0\}$. Let us denote the length set $\{|uwy| + n|vx| : n \geq 0\}$ of P_j by $\mathrm{len}(P_j)$.

Let N_w denote the number of words in L of length $< |w|$. We modify A_i by adding a modulo m counter. If $w = w_1 \cdots w_n$ is the input to A_i, and A_i has processed the prefix $w_1 \cdots w_{t-1}$, $t \leq n$, then the counter will store $N_{w_1 \cdots w_{t-1}} \pmod m$. On reading w_t, A_i increments the counter by 1 for each language P_j such that $t - 1 \in \mathrm{len}(P_j)$. The PDA A_i accepts w if and only if $N_w + i \equiv r \pmod m$. It follows that $\bigcup_{i=1}^{c} L(A_i) = \mathrm{dec}_{m,r}(L)$, as required. ■

7. Additional remarks

We point out some additional results of Rigo that are relevant. In [14, Theorem 13], he proved that if P is a polynomial that is non-negative at the natural numbers, then there exists a regular language such that extraction by the index set $\{P(n) : n \geq 0\}$ is regular. In [13, Proposition 17], he sketches the proof that extraction of an infinite regular language by the index set $I = \{2, 3, 5, 7, \ldots\}$ of primes is always non-regular.

8. Open problems

(1) Numerical evidence suggests that if $T_n = (\epsilon + (0 + 1)^*0)(1^n)^*$ (which can be accepted with an n-state DFA), then $\mathrm{even}(T_n)$ requires $(n + 2)2^{n-2} - 1$ states. Prove this and generalize to larger alphabets.
(2) Given a CFL L, is it decidable whether or not $\mathrm{even}(L)$ is a CFL?

Acknowledgments

We thank the referees for a careful reading of the paper.

References

[1] J.-P. Allouche, J. Shallit, Automatic Sequences: Theory, Applications, Generalizations, Cambridge University Press, 2003.
[2] J. Berstel, L. Boasson, The set of minimal words in a context-free language is context-free, J. Comput. System Sci. 55 (1997) 477–488.
[3] J. Berstel, L. Boasson, O. Carton, B. Petazzoni, J.-E. Pin, Operations preserving regular languages, Theoret. Comput. Sci. 354 (2006) 405–420.
[4] J.-C. Birget, Partial orders on words, minimal elements of regular languages, and state complexity, Theoret. Comput. Sci. 119 (1993) 267–291.
[5] E. Charlier, M. Rigo, W. Steiner, Abstract numeration systems on bounded languages and multiplication by a constant, INTEGERS 8 (2008) #A35.
[6] J.E. Hopcroft, J.D. Ullman, Introduction to Automata Theory, Languages, and Computation, Addison-Wesley, 1979.
[7] L. Ilie, On a conjecture about slender context-free languages, Theoret. Comput. Sci. 132 (1994) 427–434.
[8] L. Ilie, On lengths of words in context-free languages, Theoret. Comput. Sci. 242 (2000) 327–359.
[9] P.B.A. Lecomte, M. Rigo, Numeration systems on a regular language, Theory Comput. Syst. 34 (2001) 27–44.
[10] J.H. van Lint, R.M. Wilson, A Course in Combinatorics, Cambridge University Press, 1992.
[11] G. Pighizzini, J. Shallit, Unary language operations, state complexity and Jacobsthal's function, Internat. J. Found. Comput. Sci. 13 (2002) 145–159.
[12] G. Păun, A. Salomaa, Thin and slender languages, Discrete Appl. Math. 61 (1995) 257–270.
[13] M. Rigo, Generalization of automatic sequences for numeration systems on a regular language, Theoret. Comput. Sci. 244 (2000) 271–281.
[14] M. Rigo, Construction of regular languages and recognizability of polynomials, Discrete Math. 254 (2002) 485–496.
[15] J. Shallit, Numeration systems, linear recurrences, and regular sets, Inform. Comput. 113 (1994) 331–347.